Problem Sheet 7: Program Slicing
and Abstract Interpretation
Sample Solutions

Chris Poskitt*
ETH Zürich

Starred exercises (*) are more challenging than the others.

1 Program Slicing

i. Here is the program dependence graph for the program fragment (blue arrows are from the use-definition analysis; red arrows indicate control dependencies):

```
ENTRY
1 x := 0
2 y := 0
3 i := n
4 j := n
5 i > 0
6 x := x + 1
7 i := i - 1
8 j := i
9 j > 0
10 y := y + 1
11 j := j - 1
12 print(x)
13 print(y)
```

ii. For slicing criterion print(x), i.e. block 12, we get:

```
x := 0;
i := n;
while i > 0 do
    x := x + 1;
i := i - 1;
end
print(x);
```

*These solutions are adapted from previous iterations of the course when Stephan van Staden was the teaching assistant.
For slicing criterion `print(y)`, i.e. block 13, we get:

\[
y := 0;
i := n;
while i > 0 do
  i := i - 1;
j := i;
while j > 0 do
  y := y + 1;
j := j - 1;
end
end
print(y);
\]

2 Abstract Interpretation

i. We begin by mapping every variable to $\bot$ (except for $x, y$ in $A_1$, which are respectively mapped to $+ \bot$ by assumption). Then, we iteratively update the (abstract) values of variables by applying the system of equations.

<table>
<thead>
<tr>
<th>Abstract States</th>
<th>Iterations</th>
<th>Final Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_1(x)$</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>$A_1(y)$</td>
<td>$\top$</td>
<td>$\top$</td>
</tr>
<tr>
<td>$A_2(x)$</td>
<td>$\bot$ $+$</td>
<td>$\top$ $T$</td>
</tr>
<tr>
<td>$A_2(y)$</td>
<td>$\bot$ $+$</td>
<td>$+$ $T$</td>
</tr>
<tr>
<td>$A_3(x)$</td>
<td>$\bot$ $+$</td>
<td>$T$ $T$</td>
</tr>
<tr>
<td>$A_3(y)$</td>
<td>$\bot$ $+$</td>
<td>$+$ $T$</td>
</tr>
<tr>
<td>$A_4(x)$</td>
<td>$\bot$ $+$</td>
<td>$T$ $T$</td>
</tr>
<tr>
<td>$A_4(y)$</td>
<td>$\bot$ $+$</td>
<td>$+$ $T$</td>
</tr>
<tr>
<td>$A_5(x)$</td>
<td>$\bot$ $+$</td>
<td>$0$ $0$</td>
</tr>
<tr>
<td>$A_5(y)$</td>
<td>$\bot$ $+$</td>
<td>$+$ $T$</td>
</tr>
</tbody>
</table>

ii. The analysis is not very precise: it cannot prove that $y$ is positive when the program fragment completes (i.e. at $A_5$).

iii. (a) If we compute the factorial using a program that does not utilise the subtraction operator, then the result of the analysis becomes more precise:
(b) Perhaps changing the program for the analysis to work more precisely is not the best approach—let’s try to improve the analysis! We’ll try a so-called relational analysis with domain $\mathcal{P}(-,0,+ \times \{-,0,+\})$ to represent program states $(x,y)$. A relational analysis is more precise because the domain can express dependencies, or relationships, between $x$ and $y$.

We use the original version of the program fragment, but the new system of equations below:
We use the domain $\mathcal{P}(\{-,0,+,\} \times \{-,0,+,\})$ to represent the program state $(x,y)$. This is a so-called relational analysis. The relational analysis is more precise because the domain can express dependencies, or relationships, between $x$ and $y$.

$$A_1 = \{(+,\cdot), (+,0), (+,+)\}$$
$$A_2 = \{(x,+) \mid (x,y) \in A_1\} \cup \{(x,y') \mid (x',y') \in A_4 \text{ and } x \in x' \ominus +\}$$
$$A_3 = A_2 \cap \{(x,y) \mid x \in \{-,+\} \text{ and } y \in \{-,0,+\}\}$$
$$A_4 = \{(x',y) \mid (x',y') \in A_3 \text{ and } y \in x' \otimes y'\}$$
$$A_5 = A_2 \cap \{(0,y) \mid y \in \{-,0,+\}\}$$

and obtain a more precise analysis allowing us to deduce that $y$ will be positive after execution finishes:

<table>
<thead>
<tr>
<th>Iterations</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_1$</td>
<td>${(+,\cdot), (+,0), (+,+)}$</td>
</tr>
<tr>
<td>$A_2$</td>
<td>$\emptyset {(+,+)}$</td>
</tr>
<tr>
<td>$A_3$</td>
<td>$\emptyset {(+,+)}$</td>
</tr>
<tr>
<td>$A_4$</td>
<td>$\emptyset {(+,+)}$</td>
</tr>
<tr>
<td>$A_5$</td>
<td>$\emptyset {(0,+}$</td>
</tr>
</tbody>
</table>